Marching Cubes in Cylindrical and Spherical Coordinates

Jeff Goldsmith and Allan S. Jacobson
Jet Propulsion Laboratory
Pasadena, CA

January 26, 1996

Abstract

Isosurface extraction is a common analysis and visualization technique for three-dimensional scalar data. Marching Cubes is the most commonly-used algorithm for finding polygonal representations of isosurfaces in such data. We extend Marching Cubes to produce geometry for data sets that lie in spherical and cylindrical coordinate systems as well as show the steps for derivation of transformations for other coordinate systems. Such data sets arc very common in the physical sciences, and display within their natural coordinate system aids visualization considerably.

Introduction

Scalar data distributed over a three-dimensional rectangular space is a common type of science data set. Such data sets are often analyzed by finding isosurfaces in the data and rendering them as Sets of polygons. A common and fairly fast algorithm to extract such isosurfaces is "Marching Cubes" [1]. Improvements to the algorithm have been made by several researchers [2, 3].

Many data sets are distributed over non-rectangular spaces; two very commonly used spaces are. spherical and cylindrical (polar) coordinates. These are especially common in the physical sciences.

This work was motivated by the need for a multiprojection interactive isosurface visualization tool as part of the LinkWinds[4] (Linked Windows Interactive Data System) software. Currently being developed under NASA sponsorship, it is an interactive data exploration and visualization environment for quickly detecting trends, anomalies, and correlations.

Background

"Marching Cubes>' treats data as a set of voxels, each of which has eight data points as its corners. Within each voxel, a set of triangles represents the isosurface should it pass through the voxel. These polygons are shaded by picking at each vertex of the triangle a normal in the direction of' the gradient of a continuous scalar field inferred from and interpolating the original data. Since the triangles' endpoints need not lie on the original gridpoints, the authors suggest using linear interpolation of gradients at the gridpoints to determine a continuous gradient. Experience shows that this approach is adequate for most displays. This approach is coordinate-independent, but without transformation produces isosurfaces in rectangular coordinates.

Motivation

Some scientific data, while regularly gridded, is measured on a grid that is not in rectangular coordinates. In particular, Earth and astronomical observations arc usually taken in spherical coordinates. If the scale of the data is small, rectangular coordinates is a good approximation. If it is large, that leads to distortions, masking features which might be prominent in the data's natural coordinate system. Science data sets are also occasionally seen in cylindrical coordinates; Earth data sets taken over a hemisphere are often most helpful displayed in a polar map projection.

In order to produce isosurfaces of data in cylindrical or spherical coordinate systems, we have chosen to use Marching Cubes within the data's coordinate system. We later transform the resulting geometry into the display's rectangular coordinates, enabling an analyst to view isosurfaces in a variety of native coordinate systems.

Marching Cubes in Polar Coordinates

For position vectors, this is straightforward. For cylindrical coordinates, we use.

$$x' = \rho(r)\cos(\theta) \quad (1)$$
$$Y' = \rho(r)\ \sin(\theta) \quad (2)$$
$$z' = z \quad (3)$$

where cylindrical coordinates are represented as usual by $(r, \theta, z)^T$ and rectangular coordinates are represented as (z', y', $z')^T$. $\rho(r)$ is some transformation on r used to produce a map. We usually use either $\rho(r) = \cos(r)$ or $\rho(r) = \sqrt{r}$. Since all vectors are going to be column vectors (and thus post-multiply tile various matrices described below) we shall henceforth be lazy and omit the express transpose.

Transforming normals, however, is somewhat less straightforward. Normal vectors transform as the inverse transpose of the Jacobian of the coordinate transformation. Moreover, gradients in polar coordinates are not identical to those in rectangular coordinates; they are

$$\nabla_c f = \left(\frac{\partial f}{\partial r}, \frac{1}{r}\frac{\partial f}{\partial \theta}, \frac{\partial f}{\partial z}\right). \quad (4)$$

This is not much different from the ordinary gradient in rectangular coordinates, but the $1/r$ term matters.

The Jacobian of tile coordinate transformation above is

$$J(C) = \begin{bmatrix} \rho' \cos(\theta) & -\rho\sin(\theta) & 0 \\ \rho'\ \sin(\theta) & \rho\ \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (5)$$

where p' is $\partial\rho(r)/\partial r$.

The inverse transpose of this Jacobian is

$$(J^{-1})^T(C) = \begin{bmatrix} \frac{1}{\rho'}\cos(\theta) & -\frac{1}{\rho}\sin(\theta) & 0 \\ -\frac{1}{\rho'}\sin(\theta) & -\frac{1}{\rho}\cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (6)$$

To produce a normal vector in cylindrical coordinates one therefore must do the following three steps:

1. Compute the partial derivatives in the data's native coordinate system, that is, compute $(\frac{\partial f}{\partial r}, \frac{\partial f}{\partial \theta}, \frac{\partial f}{\partial z})$.
2. Compute the gradient in cylindrical coordinates, using equation 4, which is just multiplying the 6' component by $1/r$.

3. Transform to rectangular coordinates using equation 6.

This produces the following equations for normal vectors

$$N_x = \frac{1}{\rho'} \cos(\theta) \frac{\partial f}{\partial r} - \frac{1}{r\rho} \sin(\theta) \frac{\partial f}{\partial \theta} \tag{7}$$

$$N_y = \frac{1}{\rho'} \sin(\theta) \frac{\partial f}{\partial r} + \frac{1}{r\rho} \cos(\theta) \frac{\partial f}{\partial \theta} \tag{8}$$

$$N_z = \frac{\partial f}{\partial z}. \tag{9}$$

Marching Cubes in Spherical Coordinates

Using Marching Cubes in spherical coordinates is similar to using polar coordinates, but, of course, all the transformations are different and much messier. We use the spherical coordinate system components (r, θ, ϕ).

The transformation of position vectors we choose to be

$$x' = r\cos(\theta)\sin(\phi) \tag{10}$$

$$y' = r\sin(\theta)\sin(\phi) \tag{11}$$

$$z' = r\cos(\phi). \tag{12}$$

The gradient in spherical coordinates is

$$\nabla_s f = \left(\frac{\partial f}{\partial r}, \frac{1}{r\sin(\phi)} \frac{\partial f}{\partial \theta}, \frac{1}{r}\frac{\partial f}{\partial \phi}\right). \tag{13}$$

The Jacobian of the coordinate transformation is

$$J(S) = \begin{bmatrix} \cos(\theta)\sin(\phi) & -r\sin(\theta)\sin(\phi) & r\cos(\theta)\cos(\phi) \\ \sin(\theta)\sin(\phi) & r\cos(\theta)\sin(\phi) & r\sin(\theta)\cos(\phi) \\ \cos(\phi) & 0 & -r\sin(\phi) \end{bmatrix} \tag{14}$$

The inverse transpose of the Jacobian is

$$(J^{-1})^T(S) = \begin{bmatrix} \cos(\theta)\sin(\phi) & -\frac{1}{r}\frac{\sin(\theta)}{\sin(\phi)} & \frac{1}{r}\cos(\theta)\cos(\theta)\cos(\phi) \\ \sin(\theta)\sin(\phi) & \frac{1}{r}\frac{\cos(\theta)}{\sin(\phi)} & \frac{1}{r}\sin(\theta)\cos(\theta)\cos(\phi) \\ \cos(\phi) & 0 & -\frac{1}{r}\sin(\theta) \end{bmatrix}. \tag{15}$$

This produces the normal vector transformation

$$N_x = \cos(\theta)\sin(\phi)\frac{\partial f}{\partial r} - \frac{1}{r^2}\frac{\sin(\theta)}{\sin^2(\phi)}\frac{\partial f}{\partial \theta} + \frac{1}{r^2}\cos(\theta)\cos(\phi)\frac{\partial f}{\partial \phi} \tag{16}$$

$$N_y = \sin(\theta)\sin(\phi)\frac{\partial f}{\partial r} + \frac{1}{r^2}\frac{\cos(\theta)}{\sin^2(\phi)}\frac{\partial f}{\partial \theta} + \frac{1}{r^2}\sin(\theta)\cos(\phi)\frac{\partial f}{\partial \phi} \tag{17}$$

$$N_z = \cos(\phi)\frac{\partial f}{\partial r} - \frac{1}{r^2}\sin(\phi)\frac{\partial f}{\partial \phi}. \tag{18}$$

$$\tag{19}$$

Computational Issues

The main problem with these expressions occurs at poles where each geometric point maps one-to-many to data points. If the data set is good, then the many data points will be equivalent, but in practice, that is often not the case.

In polar coordinates, one cannot evaluate normals at the pole because of tile 1/r term. Either pick one of the data normals and rotate it into the appropriate position or just use "up," (0,0,1).

In spherical coordinates, the problems are similar. If $r = O$, the data mapping is many-to-one. No normal makes sense in this case, so either pick one (perhaps up) or avoid the central point of the sphere. At the North and South poles, $\sin(\phi) = O$ and we get another divide by zero. Again, wc have a many-to-one mapping of data to geometry, and a similar solution to the one used for cylindrical coordinates can be used. For the South pole, $up = $ (O, O, --1). Alternatively, one could just not evaluate exactly at the poles; pick spots some small distance away and something reasonable will be produced.

Examples

The data set used in them figures was measured by the Microwave Limb Sounder, launched aboard the Upper Atmosphere Research Satellite (UARS)[5]. It is a global map of the density of ozone in the Earth's atmosphere, and thus is a natural candidate for display in spherical coordinates.

The figures were created using the LinkWinds data visualization system. See http://twinky.jpl.nasa.gov/ for more information on LinkWinds, including how to get a free copy.

Figures 1, 2, and 3 show isosurfaces of ozone concentration in the Earth's atmosphere at a contour level of 5.478 ppm. Figure 1 is in rectangular coordinates, Figure 2 is in cylindrical coordinates (with the South pole at the center) and Figure 3 is in spherical coordinates. Figure 4 is a set of isosurfaces at different contour levels, set in spherical coordinates.

The circular region right near the South pole is one in which no data was measured. The paisley-shaped feature extending up from the South pole is the ozone hole.

Acknowledgments

The authors wish to thank Lee Elson and Joe Waters for the usc of Microwave Limb Sounder data. We extend *our* appreciation to Joe Bredekamp and Glenn Mucklow of NASA, whose Applied Information Systems Research Program (AISRP) makes this work possible. The research described in this article was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

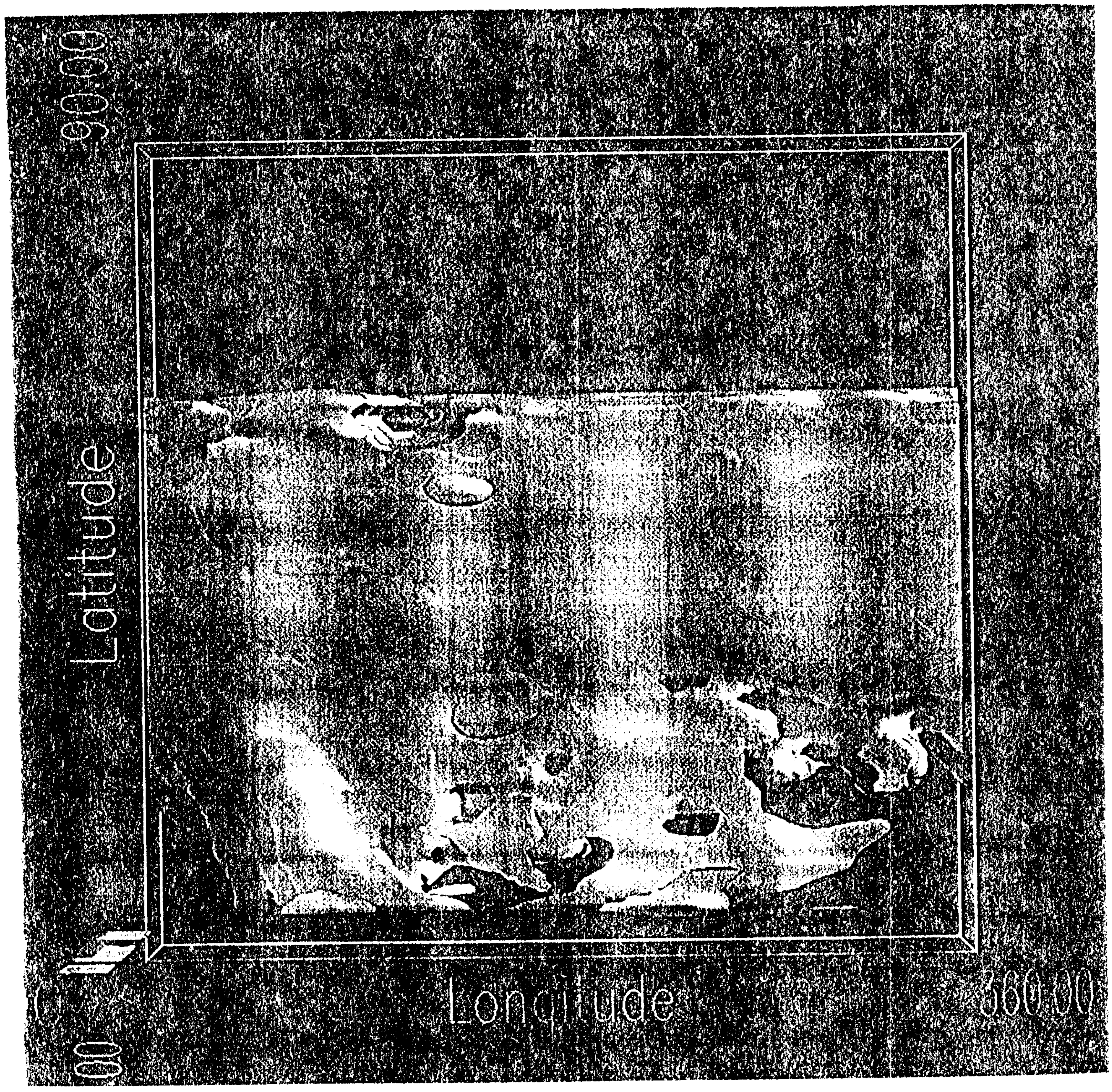
1] Lorenson, W. E., and Cline, 11. E., *Marching Cubes: A High-Resolution 3D Surface Construction Algorithm,* Computer Graphics, Vol. 21, No. 4, pp. 163-169, July, 1987.

2] Van Gelder, A., and Wilhelms, J., *7'orological Considerations in Isosurface Generation,* ACM Transactions on Graphics, Vol. 13, No. 4, pp. 337-375, Oct., 1994.

[3] Nielson, G. M. and Hamann, B., *The Asymptotic Decider: Removing the Ambiguity in Marching Cubes,* Visualization '91, pp. 83-91, 1991.

[4] Jacobson, A. S., Berkin, A. L., and Orton, M. N., *LinkWinds: Interactive Scientific Data Analysis and Visualization,* CACM Vol. 37, No. 4, PP. 42-52, April, 1994.

[5] Froidevaux, L., Waters, J. W., Read, W. G., Read, L. S. E. W. G., Flower, D. A., and Jarnot, R. F., *Global ozone observations from UARS MLS: an overview of zonal mean results,* Journal of Atmos Sci. (special issue on UARS early scientific results), vol. 51 PP. 2846-2866, 1994.


90.00
Latitude
Longitude
360.00
0.00